Multibody Simulation of REMORA CubeSat Docking to and Pushing a Spent Rocket Booster

Timothy P. Setterfield, Ryan McCormick, Junggon Kim, Rudranarayan Mukherjee
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA
818-354-9875
Timothy.P.Setterfield@jpl.nasa.gov, Ryan.L.Mccormick@jpl.nasa.gov,
Junggon.Kim@jpl.nasa.gov, Rudranarayan.M.Mukherjee@jpl.nasa.gov

***Abstract*—This paper details the multibody simulation of three phases of critical importance to the feasibility of the REMORA CubeSat space debris mitigation concept: the final approach of the CubeSat to a spent rocket booster; the grappling of the spent rocket booster using a robotic arm; and the pushing of the spent rocket booster to divert its course from another on-orbit asset. The extension of a robotic mobility and manipulation modeling toolkit (M3TK) from multibody dynamics simulation of manipulators and ground vehicles to simulation of orbital robotics is outlined. This includes the identification of the appropriate parameters required to concisely and generically describe thruster loads, thruster mixing, spacecraft control, and spacecraft navigation for the purpose of on-orbit robotics simulation. A high-level spacecraft navigator commands maneuvers to target spacecraft states. A PID spacecraft controller takes the target states and calculates desired forces and torques. A thruster mixer solves a quadratic program to determine the optimal thruster firing times for the propulsion system. Pulse width modulated actuation of eight canted cold gas thrusters is used in the simulated approach to a rocket nozzle from a distance of 200 m. A five degree of freedom robotic arm is controlled to position a pair of pincers to grasp the rocket nozzle. Contact dynamics are used to accurately simulate the grasping of the rocket nozzle by the pincers. A similar, but separate simulation is performed to assess the ability of the REMORA CubeSat to push the large spent rocket booster. This diversion maneuver makes use of an additional, larger thruster, and pushes the rocket booster in excess of 400 m. Appropriate motor control gains on the robotic arm are found to be higher during the pushing phase than those which are appropriate during free motion; this increase promotes rigidity of the arm and allows it to properly direct the pushing force. Challenges encountered in time step selection for numerical stability of the simulation are also discussed.**

TABLE OF CONTENTS

1. INTRODUCTION

Earth-orbiting spent rocket boosters are some of the largest and most dangerous pieces of space debris. The REMORA CubeSat mission concept aims to provide a cost-effective space debris mitigation service, using a 6U or 12U CubeSat built with commercial off-the-shelf components and launched as a secondary payload [1]. Once in orbit, the REMORA CubeSat would rendezvous with, dock to, track, and, if necessary, divert spent boosters and other large pieces of space debris. Of critical importance to such a mission are the final approach to and grappling of the piece of space debris. Once attached, the ability to push the space debris and divert it from a collision course with another on-orbit asset is also critical. To successfully accomplish these tasks, it is required that the joint torques stay below the maximum intermittent torques described in [1] and that the required propellant occupy only a small portion of the CubeSat.

In order to perform a realistic simulation of these three phases of the mission, the JPL M3tk (robot Mobility and Manipulation Modeling Toolkit) multibody dynamics simulation C++ library [2] was utilized. Since its inception, numerous researchers have contributed to M3tk, adding functionality that includes simulation of robotic arms, ground vehicles, contact dynamics, motor control, and more. However, prior to the work described herein, M3tk did not contain the appropriate simulation capabilities for simulating rendezvous, docking, on-orbit assembly, and space debris mitigation. This paper presents an overview of the existing M3tk methodology, details the addition of on-orbit simulation capabilities, and presents the results of using the new capabilities to simulate the critical phases of the REMORA M3tk mission concept.

2. EXISTING M3TK MODELING METHODOLOGY

M3tk allows dynamics problems to be setup using `.m3in` files that define the details of the simulated scenario including the gravity, bodies, points, joints, controllers, navigators, loop closures, joint loads, body loads, materials, and contact models. Integration properties such as the time step and total time are also specified. Generally, joints are used to specify the interfaces between different bodies at points; loads can be added at these joints or at the center of mass of the bodies. Controllers are used to exert control over the degrees of freedom of the joints, and navigators are used to plan the action of the controller at a higher level of abstraction. CAD files can be provided for the bodies as `.stl` files and, together with a file specifying the material properties of the body, used as the basis for a contact model between bodies. Figure 1 shows a graphical representation of how the dynamics problem is set up.

After definition of the dynamics problem, the simulation can be run using either `m3run` (no visualization) or `m3gv`

978-1-5386-6854-2/19/$31.00

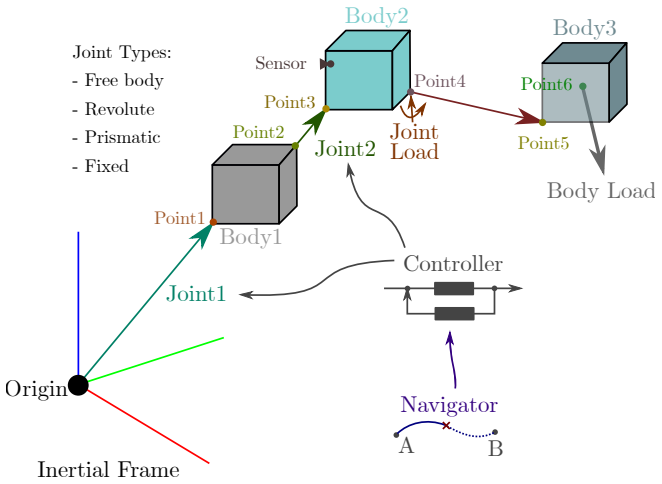


Figure 1. A topological diagram of an M3tk model.

(with visualization); the output of `m3run` can be saved for subsequent visualization. Although M3tk has a rich feature set for manipulator and ground vehicle simulation, there was previously no native support for on-orbit simulation. Thus the work herein is focused on adding such functionality.

3. Spacecraft Simulation Additions

This section provides an overview of the additional functionality added to M3tk to facilitate simulation of on-orbit operations: thrusters, mixer, spacecraft controller, spacecraft navigator, and arm navigator. A flow diagram that shows the interaction of the simulation additions in the context of the REMORA simulation is shown in Figure 2.

Thrusters

A thruster is implemented as a body load. It accepts six parameters:

`name`:	A user-specified name for the thruster
`type`:	The type of body load (`Thruster`)
`which_body`:	The body on which the thruster exerts force
`ForceMagnitude`:	The maximum magnitude of the applied body force [N] ($F_{th} \in \mathbb{R}$)
`ForcePosition`:	The position of the thruster on the body (body coordinates) [m] (${}^B\mathbf{r}_{th} \in \mathbb{R}^3$)
`ForceDirection`:	The direction of the applied force (unit vector, body coordinates) (${}^B\mathbf{n}_{th} \in \mathbb{R}^3$)

The thruster force is off by default, and is turned on using a mixer. To turn a thruster on and off, a `ForceFraction` ($\phi_{th} \in [0, 1]$) is set between 0 and 1, which is multiplied by the `ForceMagnitude` to dictate the magnitude of the body load to apply. Thus the mixer can implement either on/off thruster control (`ForceFraction` is either 0 or 1) or variable thruster control (`ForceFraction` is between 0 and 1 inclusive). The resulting body load (in body coordinates) is given as follows.

$$ {}^B\mathbf{F}_{th} = F_{th}\, {}^B\mathbf{n}_{th}\, \phi_{th} \tag{1} $$

$$ {}^B\boldsymbol{\tau}_{th} = F_{th}\, \left({}^B\mathbf{r}_{th} \times {}^B\mathbf{n}_{th} \right) \phi_{th} \tag{2} $$

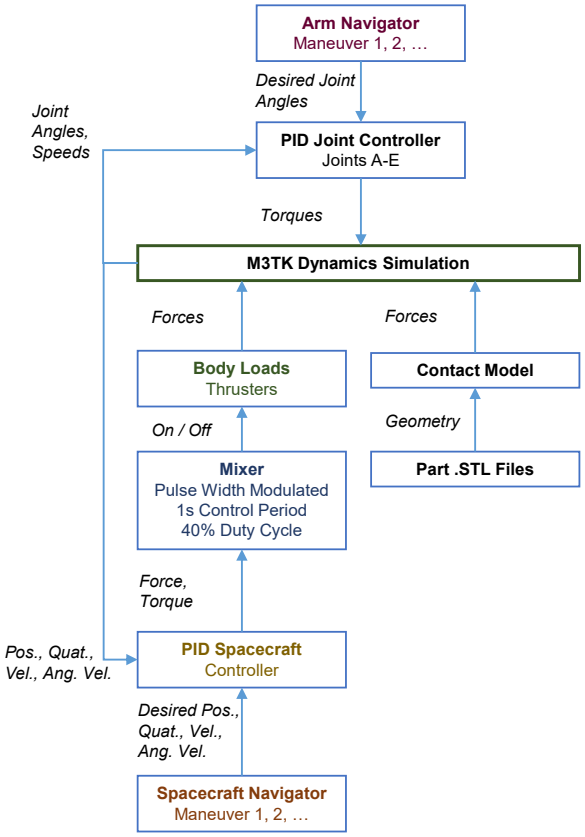


Figure 2. The flow of information through in the REMORA m3tk simulation.

Mixer

A mixer determines the thruster commands that most closely provide a desired body force and torque to a spacecraft. A mixer accepts six parameters:

`name`:	A user-specified name for the mixer
`type`:	The type of mixer to implement; currently only `PWM_THRUSTERS` (pulse-width-modulated thrusters) is supported
`num_thrusters`:	The number of thrusters to use (N_{th})
`which_thrusters`:	The names of the thrusters to use
`ControlPeriod`:	The period of the control loop [s] (t_c)
`DutyCycle`:	The fraction of the control period to allow thruster firing ($d_c \in \mathbb{R}_{>0}$)

For pulse-width-modulated control, the thruster may only be turned on ($\phi_{th} = 1$) or off ($\phi_{th} = 0$) during a fraction of the control period ($t_c\, d_c$). Herein, the on time for all thrusters is centered about the middle of this period ($(t_c\, d_c)/2$). The goal of the mixer is then to find the fractions $\{\mathbf{u}_{th} \in \mathbb{R}^{N_{th}} \mid 0 \leq u_{th_i} \leq 1 \,\forall\, i\}$ of the available firing time to open each available thruster on:

$$ \mathbf{u}_{th} = \frac{\mathbf{t}_{open}}{t_c\, d_c} \tag{3} $$

where $\mathbf{t}_{open} \in \mathbb{R}^{N_{th}}$ is the duration that each thruster is open. An average commanded force and torque of $\mathbf{u}_{F\tau} \in \mathbb{R}^6$ is sought over the duration of the control period. This can be realized by applying the equivalent impulse during the available firing time as would be applied by a force and torque

of $\mathbf{u}_{F\tau}$ over the entire control period t_c.

$$^{B}\mathbf{u}_{F\tau}\, t_c = \mathbf{M}^{+}\, \mathbf{u}_{th}\, t_c\, d_c \tag{4}$$

Here, $\mathbf{M}^{+} \in \mathbb{R}^{6\times N_{th}}$ is the pseudo-inverse of the mixer matrix $\mathbf{M} \in \mathbb{R}^{N_{th}\times 6}$, and maps thruster on/off commands $\mathbf{u}_{th}$ to commanded forces and torques $^{B}\mathbf{u}_{F\tau} \in \mathbb{R}^6$ [3]. The matrix $\mathbf{M}^{+}$ can be formed using properties of the thruster.

$$\begin{aligned} ^{B}\mathbf{u}_{F\tau} &= \begin{bmatrix} ^{B}\mathbf{u}_{F} \\ ^{B}\mathbf{u}_{\tau} \end{bmatrix} = \mathbf{M}^{+}\, \mathbf{u}_{th}\, d_c \\ &= \begin{bmatrix} F_{th_1}\, ^{B}\mathbf{n}_{th_1} & \cdots \\ F_{th_1}\left(^{B}\mathbf{r}_{th_1} \times\, ^{B}\mathbf{n}_{th_1}\right) & \cdots \end{bmatrix} \mathbf{u}_{th}\, d_c \end{aligned} \tag{5}$$

The mixer matrix $\mathbf{M} = \left(\mathbf{M}^{+T}\mathbf{M}^{+}\right)^{-1}\mathbf{M}^{+T}$ that transforms desired forces and torques into thruster commands can then be calculated.

$$\mathbf{u}_{th} = \frac{1}{d_c}\mathbf{M}\, ^{B}\mathbf{u}_{F\tau} \tag{6}$$

Thrusters can only provide force in a single direction. Therefore, for full, holonomic actuation of the 6 degrees of freedom of a satellite, 12 thrusters are required. Spacecraft are often designed with fewer than 12 thrusters, and many thruster configurations are possible. Thus a generalized optimization scheme for achieving the desired forces and torques is adopted herein.

$$\mathbf{u}^{*}_{th} = \underset{\mathbf{u}_{th},\ \text{s.t. } \mathbf{0}\le\mathbf{u}_{th}\le\mathbf{1}}{\operatorname{argmin}} \quad \frac{1}{2}\left\|\mathbf{u}_{th} - \frac{1}{d_c}\mathbf{M}\, ^{B}\mathbf{u}_{F\tau}\right\|^2 \tag{7}$$

The optimization above is solved as a quadratic program for each update of the mixer. The resultant thruster commands $\mathbf{u}^{*}_{th}$ are used to set the on and off times ($\mathbf{t}_{on}, \mathbf{t}_{off} \in \mathbb{R}^{N_{th}}$) of the thrusters. The thruster on and off times are given relative to the beginning of each control period.

$$\mathbf{t}_{on} = \frac{t_c\, d_c}{2}\left(1 - \mathbf{u}^{*}_{th}\right), \quad \mathbf{t}_{off} = \frac{t_c\, d_c}{2}\left(1 + \mathbf{u}^{*}_{th}\right) \tag{8}$$

The force fraction ϕ_{th_i} for thruster i is set to 1 for times $t_{on_i} \le t \le t_{off_i}$ and 0 otherwise.

Spacecraft Controller

A spacecraft controller determines the desired forces and torques to apply to a spacecraft given a desired position $^{W}\mathbf{r}$, attitude $^{B}_{W}\mathbf{q}$, velocity $^{W}\mathbf{v}$, and angular velocity $^{B}\boldsymbol{\omega}$. A spacecraft controller accepts up to fourteen parameters:

`name`:	Name for the spacecraft controller
`type`:	Type of the controller; either PID control, state space control, or Clohessy-Wiltshire-Hill state space control
`which_body`:	The name of the spacecraft body to control
`which_mixer`:	The name of the mixer used to apply the resulting forces and torques
`which_nav_frame`:	Inertial frame or the name of the body to use as the base navigation frame
`which_sensor(s)`:	The sensor(s) to use on the spacecraft (optional)
`init_desired_pos, quat, vel, angvel`:	The initial desired state of the spacecraft (optional)
`ControlPosition`:	Boolean – whether to control position
`ControlAttitude`:	Boolean – whether to control attitude
`PositionGain`:	For PID, nine arguments are accepted: `Kpx Kpy Kpz Kix Kiy Kiz Kdx Kdy Kdz`. For state space control, 18 arguments are accepted (the 3×6 $\mathbf{K}_{pos}$ matrix, row by row): `K11 K12 ... K31 K32 ... K35 K36`
`AttitudeGain`:	For PID, nine arguments are accepted: `Kpx Kpy Kpz Kix Kiy Kiz Kdx Kdy Kdz`. For state space control, 18 arguments are accepted (the 3×6 $\mathbf{K}_{att}$ matrix, row by row): `K11 K12 ... K31 K32 ... K35 K36`

Position and attitude control are decoupled from each other, so it is possible to have one controller that controls position and a different controller that controls attitude; it is also possible to have one controller that controls both. Commanded forces $\mathbf{u}_F$ reflect the desired position $^{W}\mathbf{r}_d$ and velocity $^{W}\mathbf{v}_d$, and commanded torques $\mathbf{u}_\tau$ reflect the desired attitude $^{B}_{W}\mathbf{q}_d$ and angular velocity $^{B}\boldsymbol{\omega}_d$. The current state of the spacecraft is obtained either from a sensor or directly from the simulation's ground truth. Filtering is not currently implemented.

The PID controller (`PID_SC`) determines the commanded forces and torques as follows:

$$\begin{aligned} ^{W}\mathbf{u}_F &= \operatorname{diag}\left(K^p_{p_x}, K^p_{p_y}, K^p_{p_z}\right)\left(^{W}\mathbf{r}_d - {}^{W}\mathbf{r}\right)\cdots \\ &\quad + \operatorname{diag}\left(K^p_{i_x}, K^p_{i_y}, K^p_{i_z}\right)\int_0^t \left(^{W}\mathbf{r}_d - {}^{W}\mathbf{r}\right)\, dt\ \cdots \\ &\quad + \operatorname{diag}\left(K^p_{d_x}, K^p_{d_y}, K^p_{d_z}\right)\left(^{W}\mathbf{v}_d - {}^{W}\mathbf{v}\right) \end{aligned} \tag{9}$$

$$\begin{aligned} ^{B}\mathbf{u}_\tau &= \operatorname{diag}\left(K^a_{p_x}, K^a_{p_y}, K^a_{p_z}\right) 2\left(^{B}_{W}\mathbf{q}^{-1}\, ^{B}_{W}\mathbf{q}_d\right)_{1:3}\cdots \\ &\quad + \operatorname{diag}\left(K^a_{i_x}, K^a_{i_y}, K^a_{i_z}\right)\int_0^t 2\left(^{B}_{W}\mathbf{q}^{-1}\, ^{B}_{W}\mathbf{q}_d\right)_{1:3}\, dt\ \cdots \\ &\quad + \operatorname{diag}\left(K^a_{d_x}, K^a_{d_y}, K^a_{d_z}\right)\left(^{B}\boldsymbol{\omega}_d - {}^{B}\boldsymbol{\omega}\right) \end{aligned} \tag{10}$$

where a p superscript indicates position gain, an a superscript indicates an attitude gain, and a 1:3 subscript extracts the vector component of a quaternion. Integration is performed numerically starting at the beginning of the simulation.

The state space controllers `STATESPACE_SC` and `CWH_SC` determine the commanded forces and torques as follows [4]:

$$^{W}\mathbf{u}_F = -\mathbf{K}_{pos}\left(^{W}\mathbf{r}^{s} - {}^{W}\mathbf{r}^{s}_{d}\right) \tag{11}$$

$$^{B}\mathbf{u}_\tau = -\mathbf{K}_{att}\left[2\left(^{B}_{W}\mathbf{q}^{-1}_{d}\, ^{B}_{W}\mathbf{q}\right)_{1:3} \quad ^{B}\boldsymbol{\omega} - {}^{B}\boldsymbol{\omega}_d\right]^T \tag{12}$$

where $\mathbf{r}^{s} = [\mathbf{r}^T \ \ \dot{\mathbf{r}}^T]^T$ is the position state.

The Clohessy-Wiltshire-Hill state space controller is identical to the generic state space controller except that the "world" navigation frame W refers to the Clohessy-Wiltshire-Hill frame. For this frame, the x-axis is the radial vector between the planet and the reference body, the z-axis is perpendicular to the orbital plane, and the y-axis completes the right-handed coordinate system. The force and torque commands $^{B}\mathbf{u}_{F\tau} = [^{B}\mathbf{u}^T_F \ \ ^{B}\mathbf{u}^T_\tau]^T$ determined by the spacecraft controller are passed to the mixer for execution.

Spacecraft Navigator

A spacecraft navigator executes a series of "maneuvers", each with a specific start time, end time, and goal state. A

spacecraft navigator accepts up to four default parameters, and up to nine parameters per maneuver.

`name`:	Name for the navigator
`which_body`:	Name of the body being navigated (same as spacecraft controller `which_body`)
`num_maneuvers`:	The total number of maneuvers to execute
`MANEUVER`:	The number of the maneuver and the start of a maneuver declaration
`which_spacecraft_controller(s)`:	Single (position and attitude) or one position-only controller followed by one attitude-only controller
`control_mode`:	Single (position and attitude) or one position and one attitude control mode to use (`DISABLE`, `DIRECT`, or `INTERPOLATE`)
`start_time`:	The start time of the maneuver [s] (t_s)
`end_time`:	The end time of the maneuver [s] (t_e)
`desired_pos`:	Desired position (optional, default `0 0 0`)
`desired_quat`:	Desired quaternion (optional, default `0 0 0 1`)
`desired_vel`:	Desired velocity (optional default `0 0 0`)
`desired_angvel`:	Desired angular velocity (optional, default `0 0 0`)

The spacecraft navigator first checks which (if any) maneuver that the spacecraft is in, based on the simulation time. At the beginning of each maneuver, it records the initial position ${}^W\mathbf{r}_i$, orientation ${}^B_W\mathbf{q}_i$, velocity ${}^W\mathbf{v}_i$, and angular velocity ${}^B\boldsymbol{\omega}_i$. If the control mode is set to `DISABLE`, the associated spacecraft controllers are disabled for the duration of the maneuver. If the control mode is set to `DIRECT`, the associated spacecraft controllers are issued a direct command with the desired spacecraft state. If the control mode is set to `INTERPOLATE`, then the spacecraft is commanded to an interpolated state between the initial and desired spacecraft state as follows.

$$ {}^W\mathbf{r}(t) = {}^W\mathbf{r}_i + \left({}^W\mathbf{r}_d - {}^W\mathbf{r}_i\right)\frac{t-t_s}{t_e-t_s} \tag{13} $$

$$ {}^W_B\mathbf{q}(t) = \text{Rot2Quat}\left({}^W_{B_i}\mathbf{R}\,\text{Exp}\left(-\text{Log}\left({}^W_{B_d}\mathbf{R}^T\,{}^W_{B_i}\mathbf{R}\right)\frac{t-t_s}{t_e-t_s}\right)\right) \tag{14} $$

$$ {}^W\mathbf{v}(t) = {}^W\mathbf{v}_i + \left({}^W\mathbf{v}_d - {}^W\mathbf{v}_i\right)\frac{t-t_s}{t_e-t_s} \tag{15} $$

$$ {}^B\boldsymbol{\omega}(t) = {}^B\boldsymbol{\omega}_i + \left({}^B\boldsymbol{\omega}_d - {}^B\boldsymbol{\omega}_i\right)\frac{t-t_s}{t_e-t_s} \tag{16} $$

Above, Rot2Quat$(\cdot)$ converts a rotation matrix into a quaternion, B_i and B_d are the initial and desired body frame locations respectively, Exp$(\cdot)$ performs the mapping so(3) $\rightarrow$ SO(3) (angle-axis vector to rotation matrix) and Log$(\cdot)$ performs the inverse mapping SO(3) $\rightarrow$ so(3).

Arm Navigator

The arm navigator executes a series of "maneuvers" on a robotic arm with N_j joints, each with a specific start time, end time, and goal state. The arm navigator accepts six default parameters and up to nine parameters per maneuver.

`name`:	Name for the arm inv. kinematic navigator
`type`:	Type of the navigator (`ARM_INVERSE_KINEMATIC`)
`which_base`:	The base body of the arm (no movement relative to the first joint)
`which_base_joint`:	The joint connecting the base to the rest of the system
`which_end_effector`:	The end effector body whose pose is being controlled
`num_maneuvers`:	The total number of maneuvers to execute
`MANEUVER`:	The number of the maneuver and the start of a maneuver declaration
`which_joint_controller`:	The `PID_NDOF` controller (from M3tk) used to control the joints
`control_mode`:	The control mode to use (`DIRECT` or `INTERPOLATE`)
`start_time`:	The start time of the maneuver [s] (t_s)
`end_time`:	The end time of the maneuver [s] (t_e)
`base_freedom`:	Freedom of the base (`FIXED`, `ROTATING`, `TRANSLATING`, or `FLOATING`) (optional, required with `desired_pos` or `desired_quat`, not with `desired_goal`)
`which_nav_frame`:	The body serving as the navigation frame for the end effector (optional, required with `desired_pos` or `desired_quat`, not with `desired_goal`)
`desired_pos`:	Desired position of the end effector (optional, default `0 0 0`)
`desired_quat`:	Desired quaternion of the end effector (optional, default `0 0 0 1`)
`desired_goal`:	Desired joint angles ($\mathbf{j}_d \in \mathbb{R}^{N_j}$) (optional, either this or `desired_pos` and `desired_quat` are required)

The functionality of the arm navigator is very similar to that of the spacecraft navigator except that instead of navigating the spacecraft, the controller navigates the end effector of a robotic arm. At the beginning of each maneuver, the navigator records the initial joint angles $\mathbf{j}_i$, the initial position ${}^W\mathbf{r}_i$, and the initial orientation ${}^W_E\mathbf{q}_i$ of the end effector. In maneuvers using the `DIRECT` control mode, the desired joint angles $\mathbf{j}_d$, or position ${}^W\mathbf{r}_d$ and/or orientation ${}^E_W\mathbf{q}$ are commanded directly. In maneuvers using the `INTERPOLATE` control mode, the joint angles, or position and/or orientation targets are interpolated. Joint angles are interpolated linearly, and position and/or orientation are interpolated as in Equations 13 and 14.

When the end effector's desired position and/or quaternion are specified, a nonlinear optimizer is used to solve the inverse kinematics and determine the appropriate desired joint angles $\mathbf{j}_d$ [5]. Different inverse kinematic constraints are placed on the system depending on the type of control used. If a desired position ${}^W\mathbf{r}_d$ is specified, then a position constraint is placed on the robotic arm's end effector; if a desired quaternion ${}^E_W\mathbf{q}$ is specified, then an orientation constraint is placed on the robotic arm's end effector. If the base freedom is `FIXED`, then a position and orientation constraint are placed on the base body; if the base freedom is `ROTATING`, then a position constraint is placed on the base body; if the base freedom is `TRANSLATING`, then an orientation constraint is placed on the base body; finally, if the base freedom is `FLOATING`, then no constraint is placed on the base body. Note that these constraints are on the inverse kinematics solution, and not on the dynamic motion of the base body itself. The existing inverse kinematic solver in M3tk is used to determine the desired joint angles $\mathbf{j}_d$. When any base freedom other than `FIXED` is specified, the inverse kinematic solver will also return a solution for position and/or orientation of the base. These values are stored for potential use in the spacecraft navigator when it is using a control mode called `ARM_COMMAND` which is not described herein.

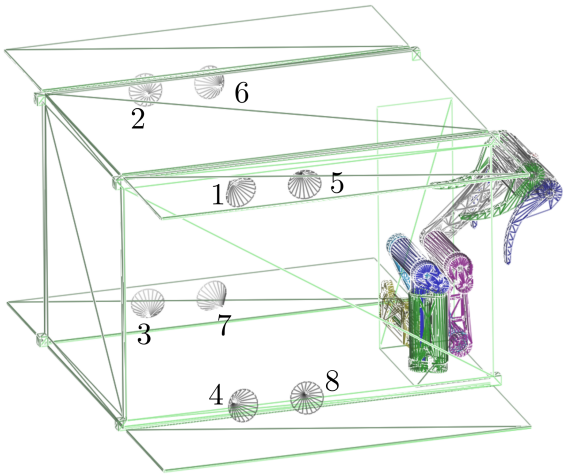


Figure 3. The REMORA cold gas thrusters and their respective thruster numbers.

When the desired joint angles $\mathbf{j}_d$ are specified directly, it is not necessary to solve an inverse kinematics problem. Finally, the resultant desired joint angles $\mathbf{j}_d$ are passed to the specified `PID_NDOF` arm controller in M3tk.

Additional Capabilities

In addition to the capabilities described above, the following sensing and absolute and relative orbital mechanics capabilities were added:

- IMU Sensor: Simulates the measurements of an inertial measurement unit.
- Planet: Used to represent a planetary body, including its mass, radius, tesseral harmonics, solar pressure magnitude and direction, and atmospheric density (absolute orbital dynamics).
- Gravity Load: Applies a body load, which includes the effects of gravity and disturbances, from an "attractor" planet or body on a second planet or body (absolute orbital dynamics).
- Orbital Reference Body: A body in a nominal, undisturbed circular or elliptical orbit; analytical equations of motion are used in its simulation.
- Relative Gravity Load: A body load simulating the relative acceleration between a body and an orbital reference body in a nominal circular or elliptical orbit (relative orbital dynamics).

A detailed explanation of these additional capabilities is excluded for the purposes of brevity, since these capabilities are not used in the REMORA simulation.

4. REMORA SIMULATION

The simulation was setup with no gravity and no orbital dynamic effects. Eleven bodies were created to enable simulation of the REMORA spacecraft: the REMORA CubeSat body (the 12U design is considered herein), the five links of the robotic arm, the end effector, the inner and outer pincers, the rocket body, and the rocket nozzle. The masses of these bodies are outlined in Table 1.

A six degree-of-freedom "Free" joint was used to attach the

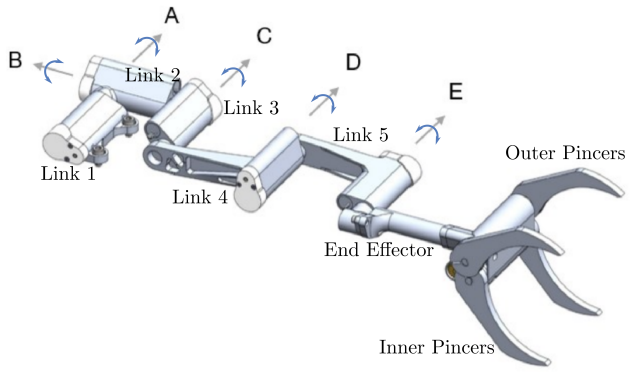


Figure 4. The REMORA arm in its unfolded configuration.

Table 1. Masses of simulated bodies. The total mass of the CubeSat is the sum of the first nine bodies; the total mass of the rocket is the sum of the final two bodies.

Body	**Mass** [kg]
REMORA CubeSat Body	22.896
Link 1	0.166
Link 2	0.157
Link 3	0.156
Link 4	0.184
Link 5	0.174
End Effector	0.218
Inner Pincers	0.040
Outer Pincers	0.048
Rocket Body	4185
Rocket Nozzle	100

REMORA body to the inertial frame. Actuation in these free-floating degrees of freedom was performed using eight 25 mN canted cold gas thrusters located on the REMORA body (see Figure 3) and one 1 N thruster with liquid propellant on the rear of the REMORA spacecraft (see Table 2 for thruster properties). Firing times for pulse width modulation were determined using a thruster mixer. Target forces and torques were provided using a PID spacecraft controller. Target spacecraft states were provided by a spacecraft navigator (see Figure 2). Ground truth spacecraft states were used in lieu of estimated values. Changes in the CubeSat's mass and center of mass due to fuel depletion were ignored.

The first link of the robotic arm was fixed in place to the REMORA body. Joints A-E from Figure 4 [1] were input as revolute joints and controlled using a five degree of freedom PID controller. Navigation was performed by an arm navigator in which the joint angles were driven directly in a rehearsed sequence (`desired_goal`) with commands interpolated between time steps (`INTERPOLATE`). The pincers were driven in the same manner using a separate PID controller and pincer arm navigator. An example `.m3in` file used for approach and grappling is shown in the appendix.

The `.stl` files for the rocket nozzle, inner pincers, and outer pincers were used to create a contact model for M3tk's built-in contact dynamics simulation. All three were set to have the material properties of steel.

Approaching

The first simulated phase of the REMORA CubeSat mission was its approach to a large spent rocket booster from a distance of 200 m. The approach was performed over a series

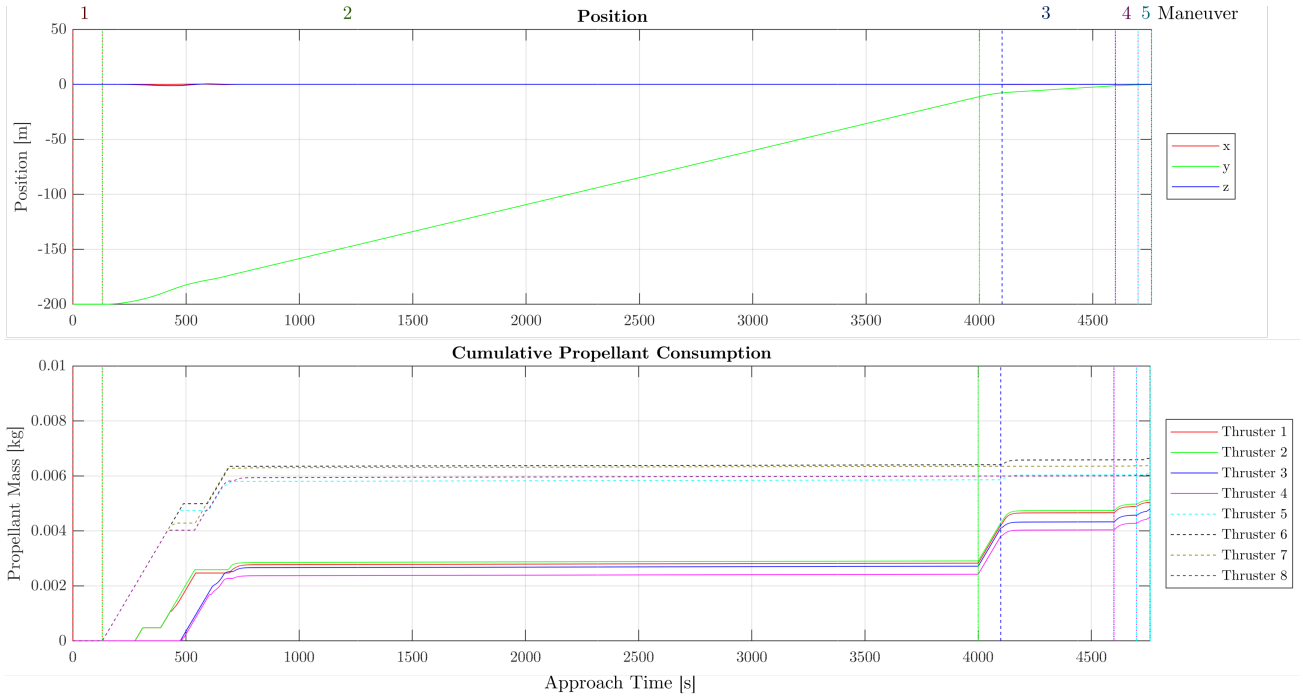


Figure 5. The position and cumulative propellant consumption during the approach phase.

Table 2. Thruster properties.

	Thrust [N]	**Specific Impulse [s]**
Cold gas	0.025	73
Monopropellant	1	220

of five maneuvers and 81.5 minutes.

Starting at a distance of 200 m, Maneuver 1 lasts 130 s and simply ensures that the REMORA spacecraft is oriented to face the spent booster. This puts the booster in view of REMORA's cameras. Herein, the spent booster is not rotating; however, if it were, techniques for mapping [6], determining the spin rate [7], and approaching the spinning booster [8] could be employed to ensure a successful approach.

The final four maneuvers, using interpolated commands (`INTERPOLATE`), bring the spacecraft progressively closer to its grappling location, at slower and slower speeds. In a real mission the transition point within each maneuver would give the operator a decision point about whether to proceed with the mission. Maneuver 2 brings the spacecraft from 200 m to 10 m in 64.5 minutes; Maneuver 3 brings the spacecraft from 10 m to 1 m in 500 s; Maneuver 4 brings the spacecraft from 1 m to 0.25 m in 100 s; finally, Maneuver 5 brings the spacecraft from 0.25 m to 0 m (its final grappling location) in 100 s. Thrusting for these maneuvers is provided by the eight 25 mN canted cold gas thrusters exclusively (i.e. the 1 N thruster is not used).

Simulation data is plotted in Figure 5. At approximately 500 s, REMORA's velocity matches the target velocity and very little propellant is consumed until approximately 400 s when the spacecraft is required to slow down. Propellant mass is calculated using the thruster firing times and assuming a N_2 cold gas thruster specific impulse of 73 s [9]. Total propellant consumption during this phase is 44.7 grams; however, no effort was made to minimize the fuel cost of this trajectory, and it is expected that the approach could be performed more efficiently.

Grappling

The second simulated phase of the REMORA CubeSat mission was the grappling of a spent rocket booster nozzle. To perform this feat, the arm navigator and pincer navigator were each given two maneuvers to complete. A 20 s arm navigation maneuver brought the arm from its stowed configuration to its ready position. Another another 20 s maneuver brought the arm to its extended position and the pincers to their ready position. Figure 7 visualizes this phase of the simulation.

Concurrently, the spacecraft navigator and controller positions the REMORA CubeSat into its grappling pose using the eight 25 mN canted thrusters; in attitude, the controller uses the same thrusters to stabilize the rotation induced by the motion of the arm. This can be seen in Figure 7 by the purple cones indicating thruster firing.

The pincers are then actuated to clamp down onto the edge of the rocket nozzle. Here, the contact dynamics produce reaction forces on the spacecraft, triggering a series of thruster openings. To maintain numerical stability of the simulation with the near-discrete contact dynamics, a short, 1 ms integration time step was used for the entire grappling simulation. This was found to be the maximum allowable time step herein; a shorter time step may be necessary in other cases.

Detailed simulation data is plotted in Figure 6. The joints experience torque spikes when arm motion is initiated or ceased. The maximum torque undergone by any joint during this phase is 5.1 mNm, which is well within the limits of the designed REMORA arm, which has a maximum intermittent torque output of 750 mNm [1]. In the displayed time window, the spacecraft approaches from -0.847 m away along the y-axis. Thrusters 1–4, which are canted toward the front, are more heavily utilized during this phase as they act to decel-

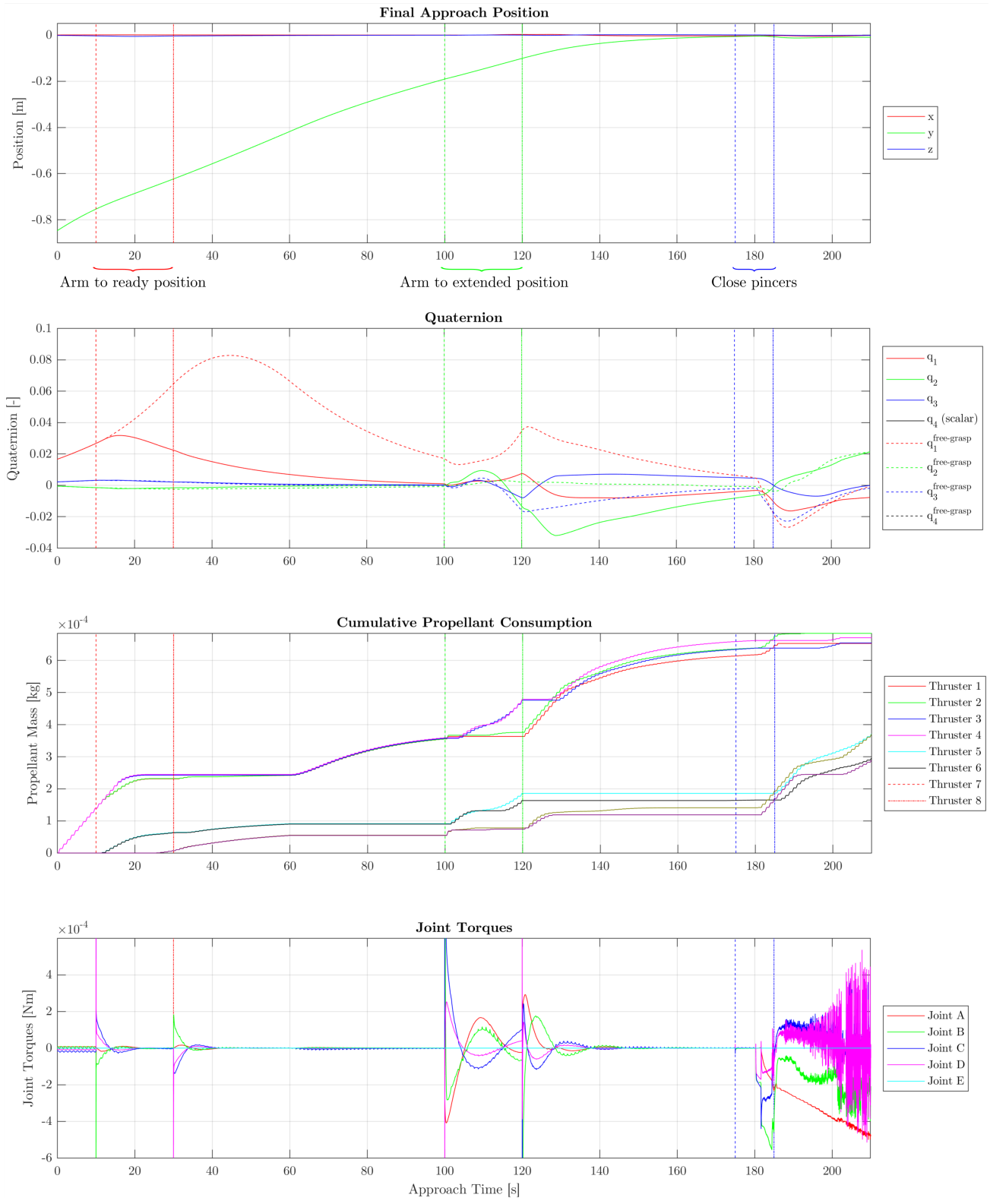


Figure 6. The position, quaternion, cumulative propellant consumption, and joint torques during the grappling phase.

erate the moving REMORA spacecraft, allowing it to slow down and commence grappling. The motion of the arm to the extended position creates an attitude disturbance requiring correction and thus consuming propellant. To illustrate the effect of attitude stabilization, an alternate simulation was run where attitude control was relinquished during the maneuvers in which the arm was moving; the result is shown by the dashed quaternion traces in Figure 6. Failure to compensate for attitude changes induced by the arm's motion causes significant deviation from the desired quaternion of 0 0 0

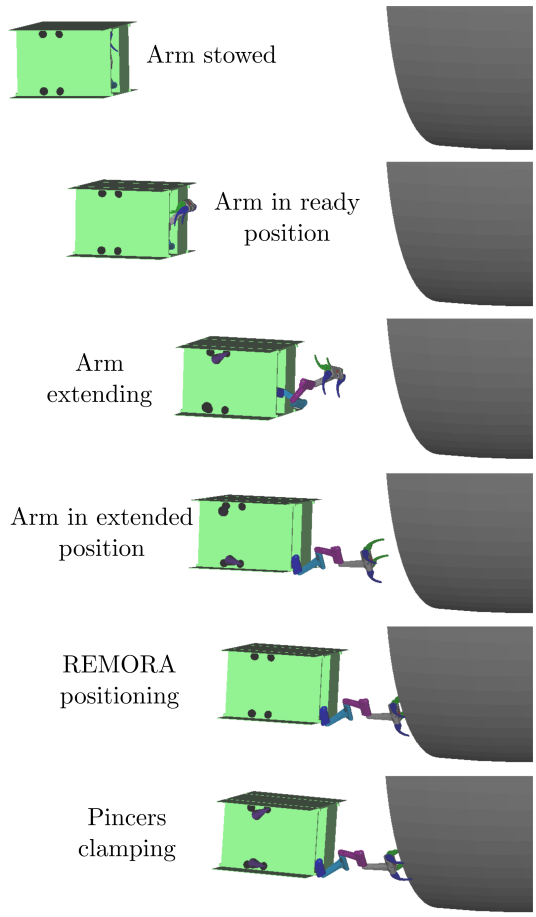


Figure 7. Visualization of the simulated grappling phase of the mission.

1. In the instance simulated, the spacecraft recovers in time and is able to complete the maneuver successfully.

The closing of the pincers creates a position and attitude disturbance once contact is made with the rocket nozzle. This is fought by actuation of the arm and thrusters (see joint torques and cumulative propellant consumption in Figure 6). In practice, the thrusters would be turned off at this point to preclude wasting fuel attempting to stabilize the satellite, and arm actuation would be used to change orientation. During the plotted period, the total propellant consumption is 4.0 grams.

Pushing

The third and final simulated phase of the REMORA CubeSat mission is the pushing of the spent rocket booster, as would be required if it is discovered to be on a collision course with an on-orbit asset. Here, attitude control was performed using the eight canted 25 mN thrusters, whereas the pushing force was provided by the 1 N rear thruster. Attitude control was performed in the frame of the rocket booster and commanded so as to point the rear thruster's force through the rocket body's center of mass.

Two maneuvers were performed. In Maneuver 1, the rear thruster's force was increased with a linear ramp over 500 s in order to reduce flexure of the arm joints. In Maneuver 2, the pulse width modulated force commands were kept at a constant level for the remaining 2.64 hrs of the simulation. To provide adequate stiffness in the arm during the pushing phase, the motor controller's PID gains were increased by an order of magnitude from those used in the grappling phase of the mission.

Initial runs of the simulation using contact dynamics from the end of the grappling phase were numerically unstable at 1 ms time steps. Instead of attempting to reduce the time step, which would drastically increase the computation time, the end effector of the arm was attached to the rocket nozzle using a fixed joint to facilitate numerically stable simulation; this did not preclude simulation of the interesting dynamics. From an initial velocity of 0 m/s, the REMORA CubeSat and attached spent rocket booster were accelerated to 0.0831 m/s, and made to travel 406.5 m over the 2.78 hr test.

An overall illustration of the pushing simulation is shown on the left-hand-side of Figure 9. The rear thruster provides a force directed at the center of mass of the spent rocket booster, and does not perform attitude control with respect to the inertial frame. As a result of flexing in the arm joints, the force from the rear thruster is not always perfectly directed through the rocket's center of mass, and the trajectory of the booster is not perfectly straight. Ideally, the rear thruster would pass directly through the trajectory of the rocket booster's center of mass (shown in blue on the right-hand-side of Figure 9). This is the case in the early stages of the simulation, but not in the later stages of the simulation where errors from the flexing of arm joints have accumulated.

Detailed simulation data is plotted in Figure 8. During the Maneuver 1 force ramp, the torques in the joints increase linearly with the applied thruster force. The y position increases exponentially as a result of the continuous application of force. The lack of attitude control with respect to the inertial frame results in a slowly drifting quaternion. A zoomed in view of joint torques shows that their values follow a sawtooth pattern, countering torques created by the pulsing of the thrusters.

The 1 N thruster, which has a specific impulse of 220 s, consumes a total of 167.0 grams of monopropellant. The cold gas thrusters consume a total of 113.8 grams of propellant attempting to keep the proper relative attitude between REMORA and the spent rocket booster. In practice, this function would be performed by using the robotic arm to vector the thrust of the 1 N thruster, and the cold gas thrusters would not consume any propellant during this maneuver. The maximum joint torque during this phase of the test was 5.3 mNm, well below the maximum intermittent torque output of 750 mNm [1].

Summary

Excluding the 113.8 grams of N_2 consumed performing stabilization that would typically be accomplished through arm actuation, 48.7 grams of N_2 and 167.0 grams of monopropellant were consumed. If the N_2 is at a pressure of 3500 psia, and thus a density of 0.28 g/cm^3 [9], and the monopropellant has a mass of 1.24 kg/L [10], the total volume of fuel required for all maneuvers herein is 0.309 L (or CubeSat units U). Since no attempt was made to minimize fuel consumption, and the diversion maneuver was very aggressive, this can be seen as an extreme upper limit to the fuel required to perform these three critical phases.

The maximum motor torque required by an arm is 5.3 mNm, far below the maximum intermittent torque output of

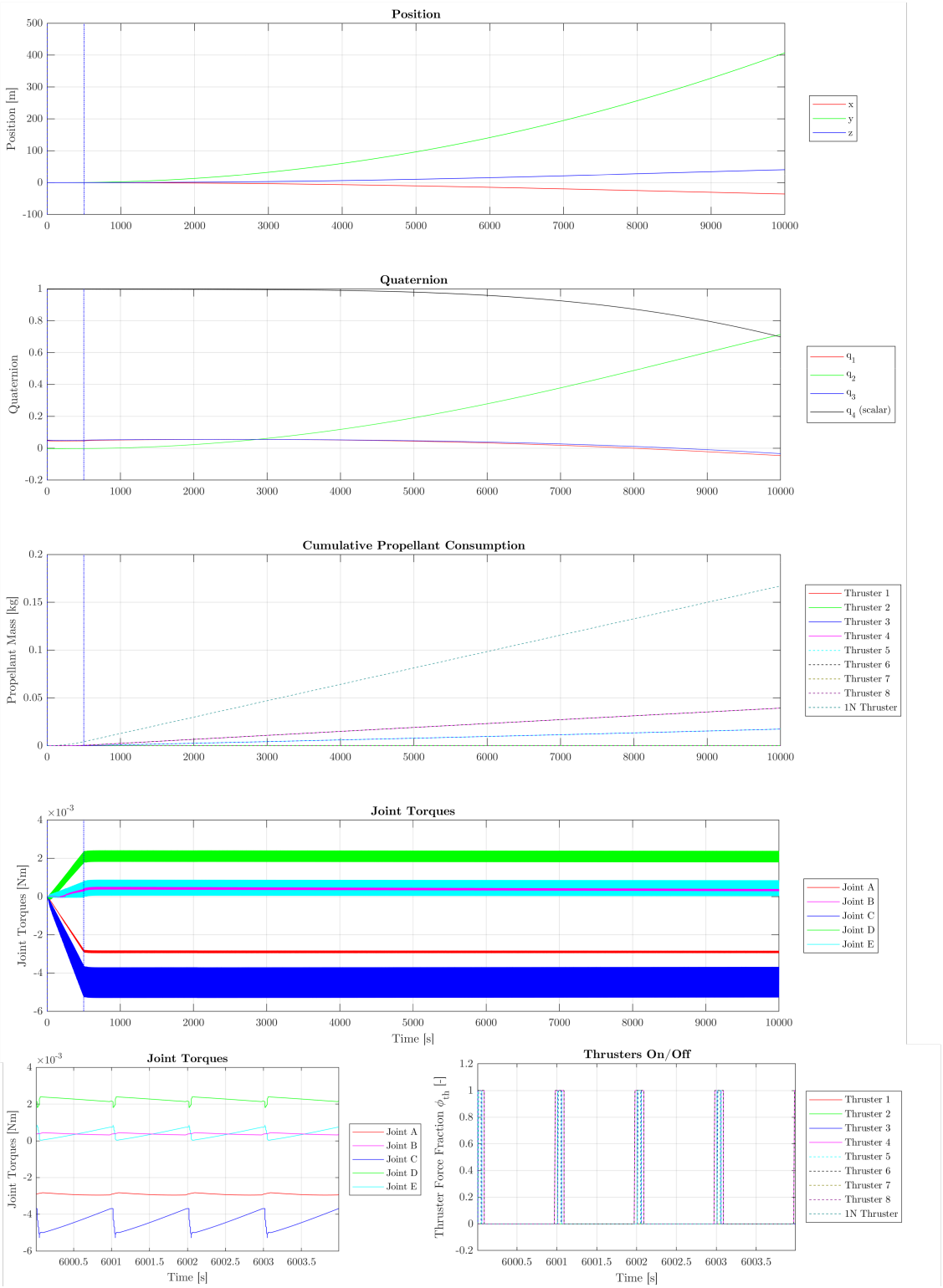


Figure 8. The position, quaternion, cumulative propellant consumption, and joint torques during the pushing phase.

750 mNm [1]. Therefore the arm, as designed, is adequate for executing the mission.

5. CONCLUSIONS

This paper outlined the extension of the existing capabilities of the M3tk multibody dynamics simulation library to enable

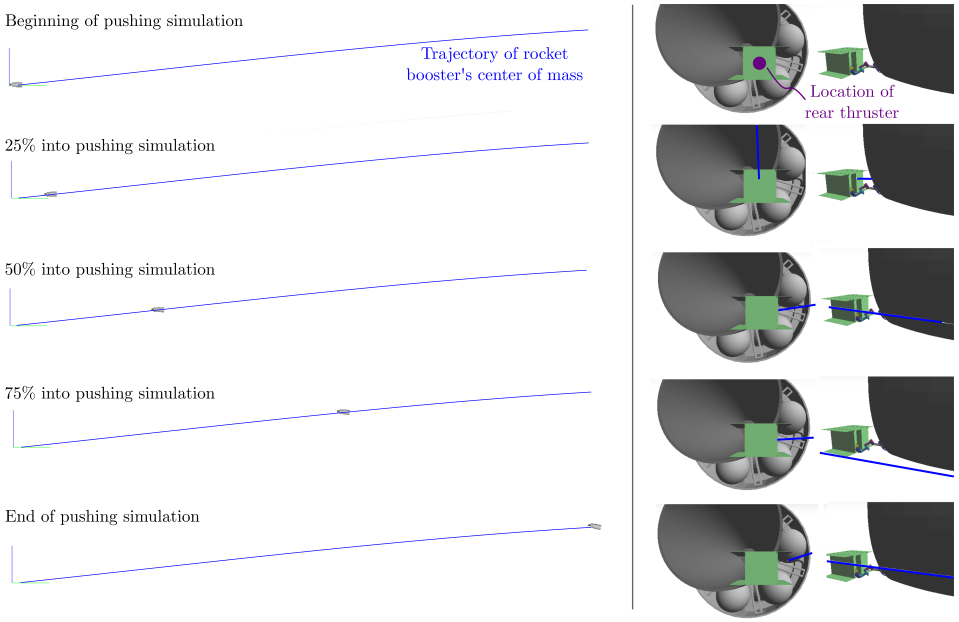


Figure 9. Visualization of the simulated pushing phase of the mission.

the testing of on-orbit assembly scenarios. The procedure utilized for adding this functionality is generic in nature and can facilitate the simulation of a variety of on-orbit proximity operations such as rendezvous, docking, grappling, and pushing.

The simulation of the critical final approach, grappling, and pushing phases of the REMORA CubeSat concept mission was presented. This simulation demonstrated the ability of a CubeSat to successfully approach a spent rocket booster from 200 m, deploy a five degree of freedom robotic arm while compensating for its motion using spacecraft control, and grapple the rocket nozzle using a pincer system. Open loop pushing the spent rocket booster in excess of 400 m over a period of less than three hours is simulated. The methods presented herein have advanced the REMORA space debris concept study by demonstrating the feasibility of the proposed proximity operations.

ACKNOWLEDGMENTS

The research described in this publication was carried out at the Jet Propulsion Laboratory, California Institute of Technology, with funding from DARPA's Tactical Technology Office (TTO) through an agreement with NASA. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. Approved for public release, distribution unlimited.

REFERENCES

[1] R. McCormick, A. Austin, K. Wehage, S. Backus, R. Miller, J. Leith, B. Bradley, P. Durham, R. Mukherjee, "REMORA CubeSat for Large Debris Rendezvous, Attachment, Tracking, and Collision Avoidance," IEEE Aerospace Conference, 2018

[2] R. Mukherjee, S. Myint, I. Kim, J. Craft, M. Pomerants, J. Kim, and L. Peterson, "M3tk: A robot mobility and Manipulation Modeling Toolkit," In International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Buffalo, 2014

[3] C. Jewison, "Reconfigurable thruster selection algorithms for aggregative spacecraft systems," SM Thesis, Massachusetts Institute of Technology, 2014

[4] B. Wie, "Space Vehicle Dynamics and Control," AIAA, https://doi.org/10.2514/4.860119, 2008

[5] J. Kim, and R. Mukherjee, "A QP-based approach to kinematic motion planning of multibody systems," In International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (pp. 38). Boston.

[6] T. P. Setterfield, D. W. Miller, J. J. Leonard, and A. Saenz-Otero, "Mapping and determining the center of mass of a rotating object using a moving observer," Int. J. Rob. Res., vol. 37, no. 1, pp. 83-103, 2018.

[7] T. P. Setterfield, D. W. Miller, A. Saenz-Otero, E. Frazzoli, and J. J. Leonard, "Inertial Properties Estimation of a Passive On-orbit Object Using Polhode Analysis," J. Guid. Control. Dyn., vol. 41, no. 10, pp. 2214-2231, 2018.

[8] D. C. Sternberg and D. Miller, "Parameterization of Fuel-Optimal Synchronous Approach Trajectories to Tumbling Targets," Front. Robot. AI, vol. 5, no. April, pp. 1-11, 2018.

[9] M. Micci, "Micropropulsion for small spacecraft.", pp. 69, AIAA, 2000

[10] A. Larsson and N. Winbord, "Green Propellants based on ADN," in Advances in Spacecraft Technologies, In-Tech, 2011.

APPENDIX

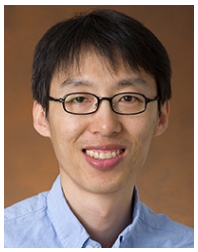
The following is an abbreviated example `.m3in` file for the approach and grappling phases.

```
**************************
DEFINE GRAVITY
**************************

Gravity : 0 0 0

**************************
DEFINE 12 BODIES
**************************

BODY    : 1
name    : Remora
mass    : 22.896
inertia : 0.290449726 0.206163465 0.301183814 0 0 0
CADPart : Remora12UNew.stl
Scale   : 0.001 0.001 0.001
Offset  : 0.003 0.01618 0.00124
Rotate  : 0 0 0 1
Color   : 100 200 100
CollisionObjectFile : NONE

BODY    : 2
name    : Link1
mass    : 0.16639
inertia : 9.857e-5 8.53e-5 2.017e-5 -1.54e-7 -1.42e-8 5.070E-6
CADPart : Link1New.stl
Scale   : 0.001 0.001 0.001
Offset  : 0.00025 -0.00948 0.03023
Rotate  : 0 0 0 1
Color   : 100 100 0
CollisionObjectFile : NONE

...

**************************
DEFINE 31 POINTS
**************************

POINT       : 1
name        : Remora_CG
which_body  : Remora
location    : 0 0 0

...

POINT       : 5
name        : Link1_Distal
which_body  : Link1
location    : 0.00025 -0.00948 0.03023

POINT       : 6
name        : Link2_CG
which_body  : Link2
location    : 0 0 0

POINT       : 7
name        : Link2_Proximal
which_body  : Link2
location    : -0.01692 -0.00589 -0.0149

...

**************************
DEFINE 12 JOINTS
**************************

JOINT             : 1
name              : RemoraFree
which_bodies      : InertialFrame Remora
which_points      : Origin Remora_CG
offset_quaternion : 0 0 0 1
type              : Free
*coordinates      : 0.71325 0.11002 0.314125 0.546882 0 -200 0
coordinates       : 0 0 0 1 0 -200 0
speeds            : 0 0 0 0 0 0 0

...

JOINT             : 3
name              : JointA
which_bodies      : Link1 Link2
which_points      : Link1_Distal Link2_Proximal
offset_quaternion : -0.5574 -0.4350 -0.5574 0.4350
type              : Revolute
coordinates       : 0
speeds            : 0
axis              : 1 0 0
gearratio         : 1

...

**************************
DEFINE 8 BODY_LOADS
**************************

BODY_LOAD       : 1
name            : Thruster1
type            : Thruster
which_body      : Remora
ForceMagnitude  : 0.025
ForcePosition   : 0.11615 -0.05282 0.10124
ForceDirection  : -0.96593 -0.18301 0.18301

...

**************************
DEFINE 1 MIXERS
**************************

MIXER           : 1
name            : Mixer1
type            : PWM_THRUSTERS
num_thrusters   : 8
which_thrusters : Thruster1 Thruster2 Thruster3 Thruster4 Thruster5
                  Thruster6 Thruster7 Thruster8
ControlPeriod   : 1.0
DutyCycle       : 0.4

**************************
DEFINE 3 CONTROLLERS
**************************

CONTROLLER          : 1
name                : ArmJointController
type                : PID_NDOF
which_joints        : JointA JointB JointC JointD JointE
control_value_type  : POSITION
kp                  : 10e-3 3e-3 8e-3 7e-3 6e-3
ki                  : 3e-4 3e-4 3e-4 3e-4 3e-4
kd                  : 20e-3 6e-3 16e-3 14e-3 12e-3
initial_goal        : 0 0 -3.5 -3.75 0

...

CONTROLLER          : 3
name                : PIDController
type                : PID_SC
which_body          : Remora
which_mixer         : Mixer1
which_nav_frame     : InertialFrame
init_desired_pos    : 0 0 0
init_desired_quat   : 0 0 0 1
init_desired_vel    : 0 0 0
init_desired_angvel : 0 0 0
ControlPosition     : true
ControlAttitude     : true
PositionGain        : 0.6 0.6 0.6 0 0 0 14 14 14
AttitudeGain        : 0.01 0.01 0.01 0 0 0 0.3 0.3 0.3

**************************
DEFINE 3 NAVIGATORS
**************************

NAVIGATOR               : 1
name                    : ArmNavigator
type                    : ARM_INVERSE_KINEMATIC
which_base              : Remora
which_base_joint        : RemoraFree
which_end_effector      : EndEffector
num_maneuvers           : 2
MANEUVER                : 1
which_joint_controller  : ArmJointController
control_mode            : INTERPOLATE
start_time              : 4650
end_time                : 4670
desired_goal            : 0 0 -3.5 -3.75 1.1

...

NAVIGATOR                   : 3
name                        : Navigator1
type                        : SPACECRAFT
which_body                  : Remora
num_maneuvers               : 5
MANEUVER                    : 1
which_spacecraft_controller : PIDController
control_mode                : DIRECT
start_time                  : 1
end_time                    : 130
desired_pos                 : 0 -200 0
desired_quat                : 0 0 0 1
desired_vel                 : 0 0 0
desired_angvel              : 0 0 0
MANEUVER                    : 2
which_spacecraft_controller : PIDController
control_mode                : INTERPOLATE
start_time                  : 130
end_time                    : 4000
desired_pos                 : 0 -10 0
desired_quat                : 0 0 0 1
desired_vel                 : 0 0 0
desired_angvel              : 0 0 0
....

**************************
DEFINE MATERIALS
materials : materials.xml
**************************

**************************
DEFINE CONTACTMODEL
contact   : 1 1
file    : remora_grasp_nozzle.m3col
**************************

************************
DEFINE INTEGRATION PROPERTIES
**************************

TIMESTEP : 0.001
DATASTEP : 1
TOTALTIME : 500

DONE
```

BIOGRAPHY

Timothy P. Setterfield *is a Guidance and Control Engineer at the Jet Propulsion Laboratory in Pasadena, CA. He holds a BScE in Mechanical Engineering from Queens University, Canada, a MASc in Mechanical and Aerospace Engineering from Carleton University, Canada, and a PhD in Aeronautics and Astronautics from the Massachusetts Institute of Technology. He has previously worked at Nanometrics Seismological Instruments developing a miniature broadband seismometer, at Carleton University developing a planetary micro-rover prototype, at the European Space Agency coordinating graduate student research in micro and hypergravity, and at MIT researching vision-based navigation using the SPHERES-VERTIGO platform.*

Ryan McCormick *received a B.S. and M.S. in Mechanical Engineering from the University of Nebraska-Lincoln in 2009 and 2011, respectively. He is a Robotics Mechanical engineer in the Robotic Vehicle and Manipulators group at JPL. At JPL, Ryan has experience with robotic manipulators and end-effectors design for Mars 2020 and JPL research tasks. Prior to JPL, Ryan designed, built, and tested miniature surgical robots.*



Junggon Kim *is a Robotics Technologist in the Robotics Modeling and Simulation group at JPL. His research interests include robot kinematics, dynamics, and motion planning. He earned his BS, MS, and PhD from Seoul National University, South Korea. Before joining JPL, he was a project scientist at Carnegie Mellon University.*

Rudranarayan Mukherjee *received a M.S. and Ph.D. in Mechanical Engineering from Rensselaer Polytechnic Institute in 2002 and 2007, respectively. He is a Research Technologist and Group Leader in the Robotics Modeling and Simulation group at the Robotics and Mobility Systems section at JPL. His primary role at JPL is to develop new technologies and find opportunities to apply them in flight missions.*